A Parametric Design Method for Computer-aided Design of Electric Machinery

D. PHILIPS, E. LOUKIPOUDIS, L. DUPRE & J. MELKEBEEK

SUMMARY In this paper the basic ideas behind an original computer-aided design (CAD) system for designing electric machinery is described. The framework in which all modules are implemented is a powerful interactive graphics system. The user thus continuously maintains control over the entire design through graphical communication. Because the CAD system is essentially conceived as a closed loop process, it supports a flexible and progressive design, allowing many trials-and-errors. Basically, the CAD system comprises two nested loops. In the outer loop, the adaptive modelling process, the geometry of the machine is controlled. The resulting geometric model maintains a dual representation of the machine. The boundary representation accepts all graphical actions of the interactive drafting system and guarantees compatibility with external systems through standards. The material region representation, on the other hand, is required for the purpose of calculation. The geometric model is hierarchically organized. This property permits the design engineer to build up the geometry from the constituent items that are relevant for the design, yielding a simple and meaningful structure. Moreover, this geometric structure need not be defined more than once. Mere assignment of new values to the parameters directly results in a new machine of the same type. Finally, any physically measurable quantity may be treated as a parameter, allowing a flexible simulation of dynamics. The inner feedback loop of the CAD system is the calculation process. In this process two basic calculation tools are provided. The magnetic network tool permits the condensed description of simple field patterns at a low cost. The more expensive finite element routine provides an accurate description of complex fields. A powerful feature of the calculation process is that it supports any combination of both tools. The calculation tools are assigned to distinct machine subregions in the discretization process. This discretization is performed upon the geometric model and thus allows re-discretization while preserving the geometry. Conversely, once a satisfactory discretization is achieved, the resulting discretized model may be embedded within the geometric model.

1. Introduction

During the design procedure of an electric machine the design engineer usually aims at the optimization of a specific type of machine with regard to one or a very reduced set of significant characteristics. This design goal may be the maximization of a perform-

D. Philips, E. Loukipoudis, L. Dupre & J. Melkebeek, Laboratory for Industrial Electricity, State University of Gent, Gent, Belgium.
ance characteristic, e.g. the torque output, or the minimization of a disturbance, e.g. the harmonic content of a phase current. These characteristics are interpreted from a physical representation of the machine, which is the result of a simulation.

The starting point for a simulation, on the other hand, is the geometry of the machine. In general, this geometry may be rather complex. Nevertheless, for the design engineer it is highly structured, due to the considerable symmetry and the numerous repetitions of similar or identical structures. As a result, the machine can be completely defined using a reduced set of design parameters. The cross-section of an induction machine, for example, is completely determined by a few radii, the type and the number of slots, and the airgap width.

We may therefore define the design procedure more precisely as the procedure in which the values of the design parameters are determined so that optimal values are obtained for the significant characteristics.

Due to the complexity of the machine, the design engineer is not inclined to perform this optimization analytically. He will rather try to reach the design goal through an intelligently controlled trial-and-error scheme. The design engineer will thus simulate the machine for different values of the design parameters. After each such simulation, he evaluates the significant characteristics with respect to the design goal and adjusts the design parameters for the next iteration accordingly. Consequently, we may consider the design procedure as a closed loop process, interactively controlled by the user. Repetitively cycling through this inspect-and-adjust process, the design engineer progressively proceeds towards the design goal (Fig. 1).

Unfortunately, commonly used contemporary computer-aided design (CAD) packages somehow confine the user to follow on a priori defined design scheme, although they may offer an extensive set of powerful tools to flexibilize this scheme.

Typically, in this scheme the successive stages of definition, preprocessing, calculation and postprocessing must be executed sequentially. Moreover, the powerful graphical communication between the user and the system is rarely exploited to its limits. Whereas computer graphics are used to render the definition process before the simulation and the interpretation afterwards more user-friendly, the user looses this communication during the actual simulation itself.

Therefore, we introduce a different approach in building a CAD system for electric machinery in this paper. The nucleus of the CAD system we present is an interactive graphics system, in which all the actions to be undertaken during the design procedure are implemented.

Rather than following a straight path, we allow the user to create his own design procedure based on the design parameters and leading to an intermediate design goal.
The design procedure itself is defined in a user-friendly environment through interactive computer graphics techniques.

Once defined, the user is free to execute the design procedure as many times as needed, varying the values of the design parameters and acquiring different results. Moreover, an entire design session can be defined as a higher level design procedure, in which a user-defined design procedure is automatically re-executed until a goal is reached.

The implementation of this philosophy in its most condensed form reduces to the structure in Fig. 2. The geometric model contains the complete set of data required to define the machine. In the geometric modeller this model is repetitively modified by the user during the design session, which we therefore have called the adaptive modelling process. This modelling process is discussed in the first part of the paper.

![Diagram](image)

**Fig. 2.** The adaptive modelling process.

The data in the geometric model is organized so that it can be directly interpreted in the calculation process, which is the subject of the second part of the paper. The physical model produced by this calculation process describes the behaviour of the machine.

2. The Adaptive Modelling Process

2.1 The Geometric Modeller

The geometry of a cross-section of any electric machine is built up using an interactive drafting system. This graphic system indeed provides the full set of basic commands to construct the geometry in a flexible way, such as drawing, copying, transforming and modifying graphic primitives. Since the geometrical primitives thereby obtained are lines and curves, the resulting geometry is defined as a collection of linear boundaries. We identify this collection as the *boundary representation* of the cross-section.

In order to obtain a physically consistent model of the machine, however, we need the cross-section represented as a two-dimensional material space. Moreover, this two-
dimensional space should be comprised of different material regions satisfying two rules: any two material regions do not overlap and the set of all material regions defined covers the whole space within the outer boundary of the electric machine. This partitioning of the geometry forms the material representation, which is a higher level of structuring the pure geometrical data.

Having the boundary representation of the geometry mentioned above, we allow boundaries to be created using all the facilities the contemporary general purpose CAD systems provide, including transferring drawings from different CAD packages through graphic exchange standards (e.g. IGES). Moreover, such an interface can be used for passing the required information to NC machines for manufacturing the design solution. The process of using the interactive graphic facilities to create the boundary representation of a cross-section we call the geometry definition process. It is the major part of the geometric modeller.

The material definition process that follows in fact converts the representation of the drawing being a set of geometric boundaries into a set of two-dimensional regions. The resulting structure (Fig. 3) is similar to the B-representation models in the three-dimensional Solid Modellers [1].

![FIG. 3. Permanent magnet rotor with its geometrical representation.](image)

In order to maintain the consistency of the region representation as a partitioning of the geometry we provide special operations within the geometric modeller to be performed on the boundary representation. An automatic checking against region overlapping is implemented by intersecting the boundaries comprising the different regions. The undefined-space test can also be performed on the boundaries checking whether every boundary but the outer one is a member of exactly two material regions.

The region representation of the cross-section also incorporates some physical knowledge about the materials used. Thus to every material region a material property is associated. Moreover, any set of numerical parameters that determine such a property can be treated as design parameters, too. This gives the opportunity to optimize the materials used keeping the geometric shape intact.

The integration of both boundary and material representation in one geometric model is essential for the reason that the interactive graphic tools should also treat the material regions as graphic objects. For example, it is feasible to pick regions with the graphic input device available; however, this picking must not be performed on the level of the conventional graphic primitives, because a primitive usually belongs to two material regions, thus making the selection ambiguous.

So far we have introduced the material region representation of the electric
machine, together with the specific operations we apply to it. Moreover, we have integrated this region representation with the conventional boundary representation of the geometry. We thus have structured the geometrical data in an appropriate form for further processing, while preserving both the full power of the drafting system and the compatibility with standards. In order to make the design procedure really flexible, however, we should structure both representations into a parametrized model upon which the design procedure acts.

2.2 The Parametrization: Argumentation

The main purpose of the geometric modeller is indeed to provide the means for organizing the geometry in the entities the engineer deals with when designing an electric machine. At any stage of the design procedure the engineer thinks of the motor as being comprised of teeth, slots, yoke, etc., rather than being built up of lines and curves. Thus, it is quite convenient to structure the geometry as composed of these constituent items, each represented by a set of graphic primitives. These constituent items can easily be classified into types (or families of parts), so that all the items of the same type have a similar geometrical structure and occupy a specific place in the objects' hierarchy. This property allows us to associate a parametrized model to the type. Assigning a set of values to the parameters used then produces an instance of the type.

As an example, we have depicted several types of slots in Fig. 4. Apparently, a slot is completely defined if we specify the values of the opening, width, depth, offset and fillets. Defining these quantities as parameters rather than fixed values, the geometrical structure of a type need be created only once. Figure 4 shows instances of three such slot types.

Fig. 4. Instances of three different slot types.

The geometry of the whole machine is then comprised of instances of different types. These instances are produced for a set of actual values of the parameters of the
types used. The whole collection of parameters of all these types we identify as the design parameters of the geometry. Thus we define the geometric model as a hierarchical structure of types with their parameters (Fig. 5). For a set of actual values of these parameters a fixed geometry is then produced as an instance of this hierarchical structure.

Due to the parametrization method applied, we call this structure the procedural description of the designed object.

It is obvious that an identical geometry can be achieved using different procedural descriptions with different design parameters. To start with, for some application it may be convenient to have different sets of parameters describing the same type of constituent item used.

For example, we may specify the opening between the teeth rather than the airgap chord length of the tooth's tip, though either can be used as a parameter of any tooth type. On the other hand, the same geometry can be created using completely different constituent items. For instance, a stator can be constructed out of teeth as in example 1.

```plaintext
type Stator (Rad,Xc,Yc,Nteeth, <tooth_parameters>)
OuterBoundary = circle(Xc,Yc,Rad)
SingleTooth = tooth(<tooth_parameters>)
InnerBoundary = connect_circular(SingleTooth,Xc,Yc,Nteeth)
define_material(IRON,OuterBoundary,InnerBoundary)
end.
```

**Example 1.** Procedural description of a stator built up from teeth.

However, an identical geometric result can be achieved using slots as basic constituent items.

```plaintext
type Stator(Rad,Xc,Yc,Nslots, <slot_parameters>)
OuterBoundary = circle(Xc,Yc,Rad)
SingleSlot = slot(<slot_parameters>)
InnerBoundary = connect_circular(SingleSlot,Xc,Yc,Nslots)
define_material(IRON,OuterBoundary,InnerBoundary)
end.
```

**Example 2.** Procedural description of a stator built up from slots.

From the foregoing, it is quite clear that a flexible CAD system should never use fixed constituent items, because the engineer must have the opportunity to organize the geometric model according to the specific requirements of his application. So we have introduced parametrized graphic objects instead, with their implementation and parameters definition tailored to the particular design strategy needed.

For defining parametrized objects in an electric machinery CAD system, certain requirements should be taken into account. The variation geometry approach [2, 3] in parametric design, though being very user-friendly, is not applicable because the items like the rotor or the stator have parameters which do not have metric interpretation. These are the number of slots or teeth for each of the items. Thus, by varying the number of slots, for example, instances with a different number of characteristic points will be produced. Such instances cannot be described by a single variational geometry family of parts [4]. On the other hand, the use of complex parametric models [5, 6] is not justified for the restricted set of geometrical components in an electric machine.
drawing. We therefore aim at applying the procedural approach, so that both geometrical data (families of parts) and discretization and calculation procedures are parametrized.

Further on we describe how the philosophy to structure the geometric model parametrically is implemented using a procedural parametric language [7].

2.3 The Parametrization: Philosophy

The central idea of the parametric design is that every type of constituent item is implemented as a parametric language program. The geometric modeller then acts as an interpreter whenever such a parametric graphical object is to be included in the model.

The parametric language in fact is a high-level programming language the statements of which are all commands of the geometric modeller that can be invoked interactively. The arguments of these commands are the variables of the language. Both numeric and graphic types of variables can be used so that graphic parameters can be defined for the parametric object itself.

\[
\text{type Stator}(\text{Rad}, \text{Xc}, \text{Yc}, \text{N}, \text{ToothOrSlot}, \text{Material}) \\
\text{OuterBoundary} = \text{circle}(\text{Xc}, \text{Yc}, \text{Rad}) \\
\text{InnerBoundary} = \text{connect\_circular}(\text{ToothOrSlot}, \text{Xc}, \text{Yc}, \text{N}) \\
\text{define\_material}(\text{Material}, \text{OuterBoundary}, \text{InnerBoundary}) \\
\text{end.}
\]

Example 3. Stator built up from a variable constituent item.

The graphic parameters are extremely useful particularly when a high-level constituent item is defined that is to be irrelevant of the basic items it is comprised of (Example 3).

In this example the value of the graphic parameter 'ToothOrSlot' can be passed by the main routine 'Motor' when calling the routine 'Stator'. This gives the designer the opportunity to change the type of slot or tooth used without altering the hierarchical structure of the geometrical model itself.

As shown in Fig. 5 the geometric model is a hierarchical structure on several levels. In order to create this structure with proper passing and inheriting parameters of the nested constituent items the engineer should build the geometric model writing programs rather than interactively drawing its geometry.

Unfortunately, writing parametric language programs is far from a user-friendly

![Fig. 5. The geometric model.](image-url)
interface to an interactive graphic system and certain programming experience is needed to make an efficient use of this facility. Moreover, while creating a parametric object the designer should not violate any rule upon which the parametric language is based. This can only be checked when the parametric program is interpreted by the geometric modeller.

In order to solve these problems we have chosen the interactive parametrization method [8, 9] for the creation of the types of constituent items.

2.4 The Parametrization: Implementation

The basic idea of the method is that parametric language programs defining new types of constituent items (teeth, slots, rotors, stators, etc.) are automatically generated in the course of the interactive graphic creation of an instance of these items. The commands invoked for creating these elements then are written to a file in a protocol-like style. The filing of the command sequence is done so that any data for the commands selected can be declared as variables and thus can be altered when the same command sequence is re-executed. A mechanism for defining and managing these variables has been implemented so that they can naturally serve as the set of parameters of the parametric object designed.

Essential to this approach is that a whole family of objects is created during the interactive constructing of a specific instance of it. This instance is drawn on the screen so that the designer has the full feedback and any design error can be easily detected. Thus, the parametric program generated is both syntactically and semantically correct.

The nesting of parametric modules is easily achieved by invoking parametric objects (such as teeth or slots) when interactively creating the parametric description of the higher level ones (such as rotors and stators). In this way the hierarchy of the motor constituent items is naturally embedded in the geometric model. The geometric model itself is redundant combining two different representations of the cross section. The first one is the procedural (hierarchical) description of the electric machine as a structured set of parametric language programs and the second one is the geometrical data for a certain set of actual values of the design parameters (Fig. 5).

Summarizing, we have achieved a geometric model of the machine which is organized in a hierarchical way. The use of constituent items as they are encountered in practice makes the resulting structure simple and meaningful to the user. Simultaneously, the classical boundary representation of the geometry is maintained, thereby preserving the classical operations on graphical primitives and the compatibility with standards. Moreover, for each type of machine, the structural definition is required only once. Due to the parametrization, any instance of the so defined type can be created by merely assigning new values to the parameters.

As will be explained later, the parametrized model needs not be restricted to the geometrical data only. The procedural approach allows any data describing the machine such as the electrical excitation or the mechanical motion to be integrated in the parametrized model in a flexible way. That is why we have called the most outer loop of the design process (Fig. 2) the adaptive modelling process. Embedded within it is the calculation process, described below.

3. The Calculation Process

3.1 Accuracy versus Time-efficiency

In the previous part the definition of the machine has been described. The resulting
data in the geometric model is organized in an appropriate way for further processing, which will be discussed now.

In the calculation process the electro-mechanical behaviour of the machine is simulated. The obtained results are represented in the physical model, from which the significant characteristics can be derived (Fig. 2).

Due to the closed loop structure of the design process, the calculation is to be executed quite frequently. Indeed, even in the case where the significant characteristics can be derived from one magnetostatic field calculation, the calculation process is to be executed for every point in parameter space the user wants to simulate. It is quite clear that the entire procedure can only be performed in an acceptable run-time if the time required to calculate one single field is restricted to a minimum.

Moreover, in many cases the significant characteristics cannot be derived merely from a static field calculation, but require a dynamical simulation. Commonly, such dynamic simulation is based upon a sequence of static fields, corresponding to a sequential set of time instants. The current distributions exciting the magnetic field, and the relative position of rotor and stator then are governed by dynamical relations. It is important to notice that we may treat these current excitations and the position in the same manner as the design parameters, so we can incorporate them in the parametrized model of the machine. Moreover, the closed loop structure of the adaptive modelling process allows us to update the values of the excitations and the position automatically. This feedback loop then is written as a parametric procedure (Example 4).

```
type iterate_dynamics (omega0,theta0,tstep,tmax)
  i=0.
  t=0.
  omega=omega0
  theta=theta0
  while t<tmax do
    Calculate_Static(theta,i)
    Extract_Electric (L,e)
    Extract_Torque (Tmotor)
    i=i+(v-R.i-e)*tstep/L
    omega=omega+(Tmotor-Tload)*tstep/(Jmotor+Jload)
    theta=theta+omega*tstep/2
    t=t+tstep
  enddo
end.
```

Example 4. Procedural implementation of the simplified model.

Moreover, there is no fundamental reason to restrict the quantities to be fed back to the position and current values. Indeed, any physical quantity that can be extracted from the physical model may be incorporated in the parametric procedure. This allows the user to emulate any realizable closed loop control scheme of an electric machine directly.

Once again, it is quite clear that such a dynamic simulation is feasible only if we reduce the time required for the calculation of one single field to a minimum. Apart from the apparent need for time efficiency, however, the accuracy required from the calculation process may be considerable, because some significant characteristics are extremely sensitive to this accuracy.
The foregoing remarks clearly show that any calculation process embedded in the design procedure should meet two somewhat conflicting requirements, i.e. accuracy and time-efficiency.

3.2 The Discretization: Philosophy and Implementation

Since a principal fraction of the machine cross-section comprises strongly non-linear materials, such as the motor iron, the calculation of the magnetic field inside these materials inevitably follows an iterative scheme. The total time cost for the calculation of the field may therefore be taken as the cost per iteration times the number of iterations. Experiments show that on one side the number of iterations required to calculate the non-linear field strongly depends on the 'hardness' of the non-linearity. Unless we modify the material characteristics of the machine under study, we have to accept this cost.

The effort to be paid each iteration on the other side principally involves the definition and solution of a large system of equations. The time cost of this job clearly depends on the number of unknowns. Consequently, the user will control the total time cost of influencing the number of unknowns used to describe the field.

Therefore, we provide the user with two fundamental calculation tools. The magnetic network tool, to start with, describes simple field patterns in a condense manner, thereby reducing the number of unknowns considerably. The finite element tool, on the other hand, allows the accurate calculation of complex field patterns, evidently at the expense of a higher number of unknowns. The basic philosophy underlying the calculation process now is that the calculation of the field can be based on any combination of both tools. This philosophy enables the user to shift the compromise between accuracy and time-efficiency according to the requirements of the actual simulation. For the justification of the 'coupling' of both numerical methods, we refer to ref. [13].

The above philosophy is implemented in the Discretization process (Fig. 6). Starting from the complete geometric model of the machine, the user divides the cross-section into subregions, which either a finite element calculation tool or a magnetic network tool is assigned to. Obviously, the optimal assignment of these tools strongly depends on the type of machine under study and on the actual design requirements. In order to allow a flexible achievement of this optimum, the calculation process as well has been conceived as a closed loop, see Fig. 6.
This inner feedback loop then allows re-discretization, while, however, preserving the geometrical representation. Conversely, after a satisfactory discretization has been achieved for a given machine, it may be useful to associate this discretization to the specific type this machine belongs to. Therefore, the option is offered to embed this discretization within the fully parametrized geometrical model. Consequently, in later stages the assignment of new values to the parameters does not merely create an entirely new geometry; it provides us with a complete calculation ready model as well. Figure 7 shows the cross-section of a permanent magnet motor the two tools have been applied to.

![Figure 7](image)

**Fig. 7.** The discretization of a permanent magnet motor.

4. **Concluding Remarks**

As a starting point, we have shown that the design of an electric machine essentially follows an iterative scheme, interactively controlled by the user. The basic philosophy underlying a CAD system suitable for designing electric machinery should naturally reflect this iterative scheme.

In the CAD system presented in the above presentation, this philosophy is implemented as a double feedback process. Figure 8 shows a simplified flowchart of the complete CAD system, in which both the adaptive modelling process and the calculation process can be distinguished.

The explicit separation of both processes, although not always evident, was maintained to the very end of the paper for the sake of presentation. Indeed, as more discretizing actions are progressively embedded within the model of the machine itself, both processes will obviously merge.

However, it remains essential that the CAD system is basically conceived as a closed loop process, interactively controlled by the user. Therefore, it will perfectly fit
as the environment in which a machine is designed, as the basic nature of both the environment and the design process coincide.

**Acknowledgement**

The work presented in this paper was fully supported by the Interuniversity Attraction Pole Program (IUAP no. 14) of the Belgian Government.

**REFERENCES**


